Prometheus Operator:

"Out-of-the-Box" Monitoring As A Code

Щербаков Станислав Николаевич





About Me

Профессиональные навыки

- 25 лет уверенный пользователь ПК
- Более 10 лет работы по специальности
- ~5 лет из них DevOps Engineer

Хобби

- INFOSEC
- Reverse engineering
- Console hacking



t.me/xSTASiANx

About Plesk

- 400k инсталляций продукта
- 6% сайтов в интернете (NetCraft)
- Более 20 лет на рынке
- 150+ классных инженеров

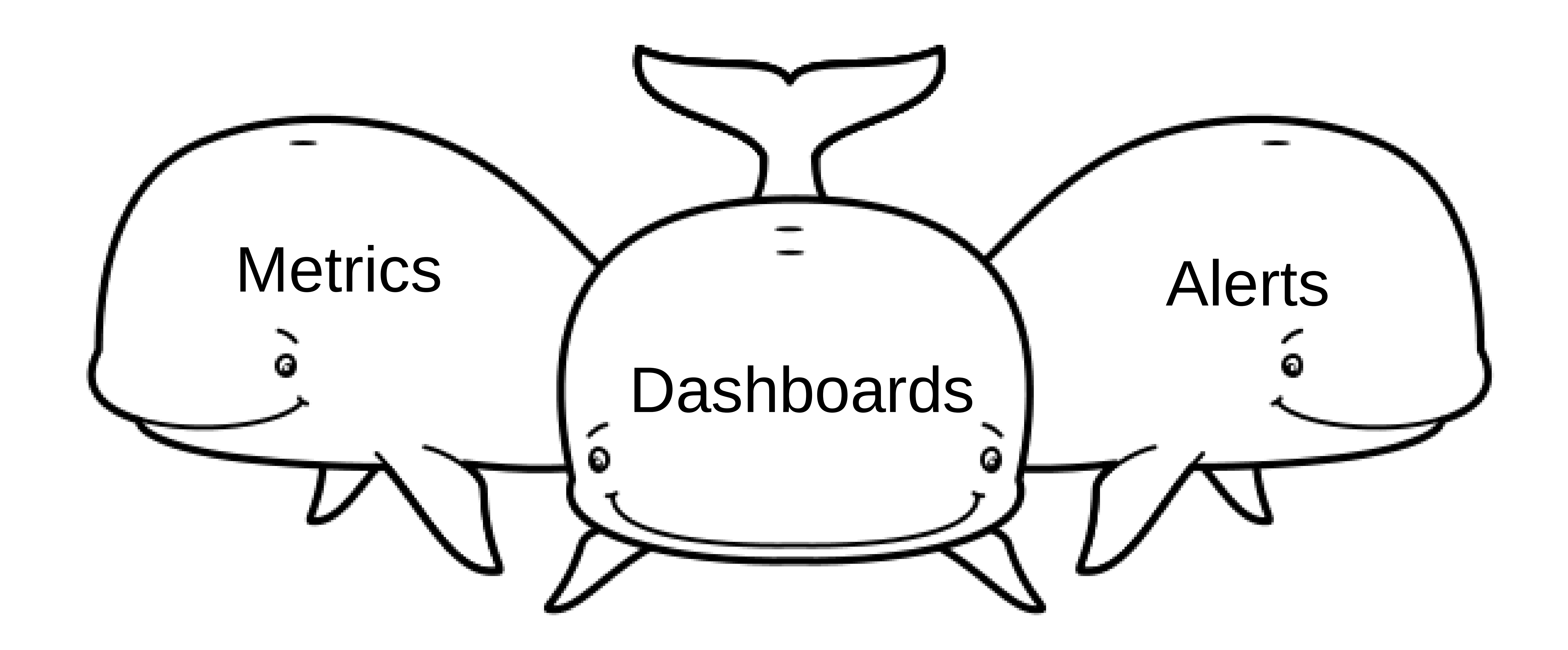


Table of contents

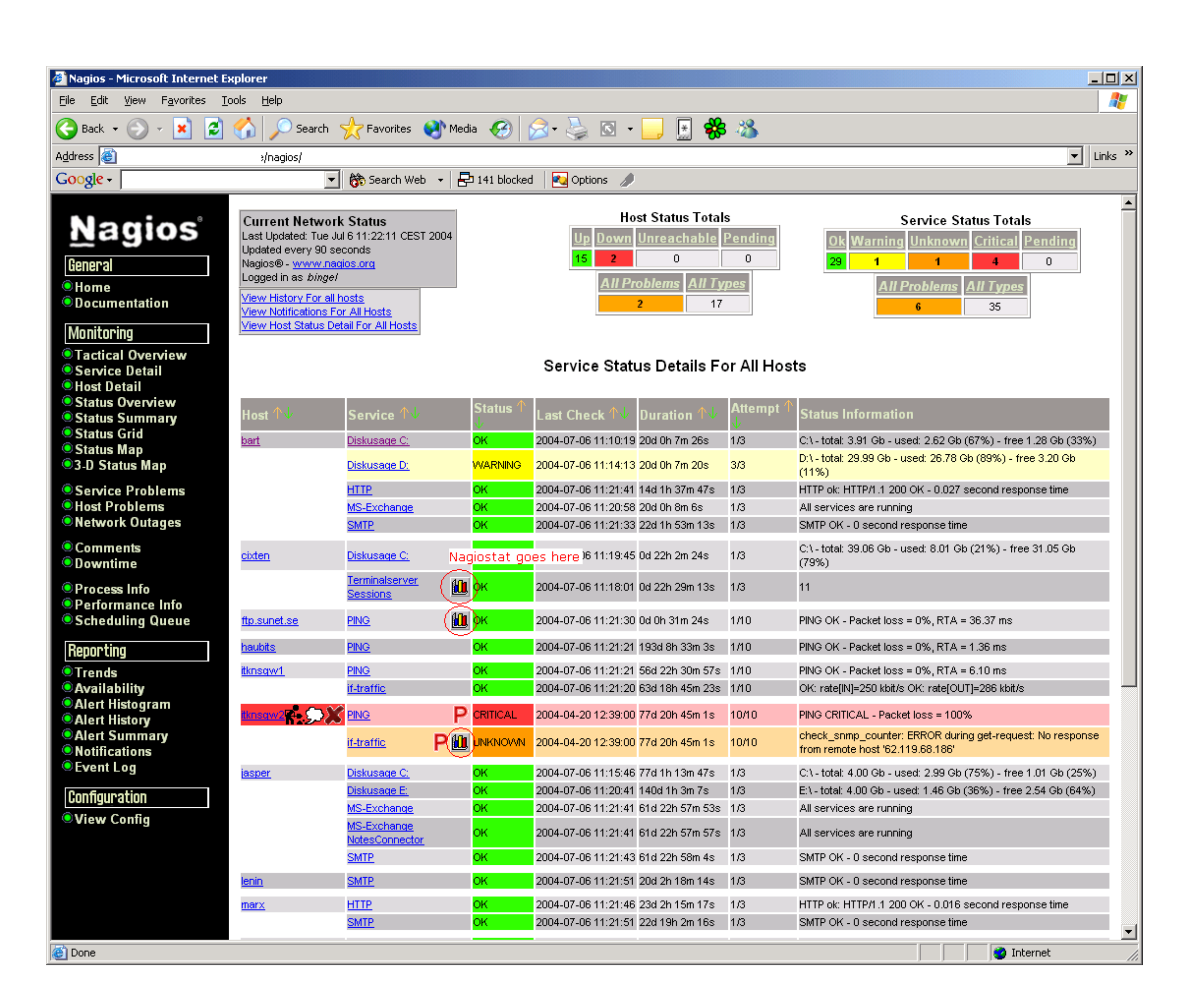
Now we are here

- The way from Legacy monitoring systems to Modern
- Monitoring Problems
- Legacy Infrastructure Service Discovery
- Prometheus Operator as the Solution
- PaaS Alerting and Dashboards — Monitoring As A Code
- Upgrades and Incidents
- Conclusion

Monitoring systems

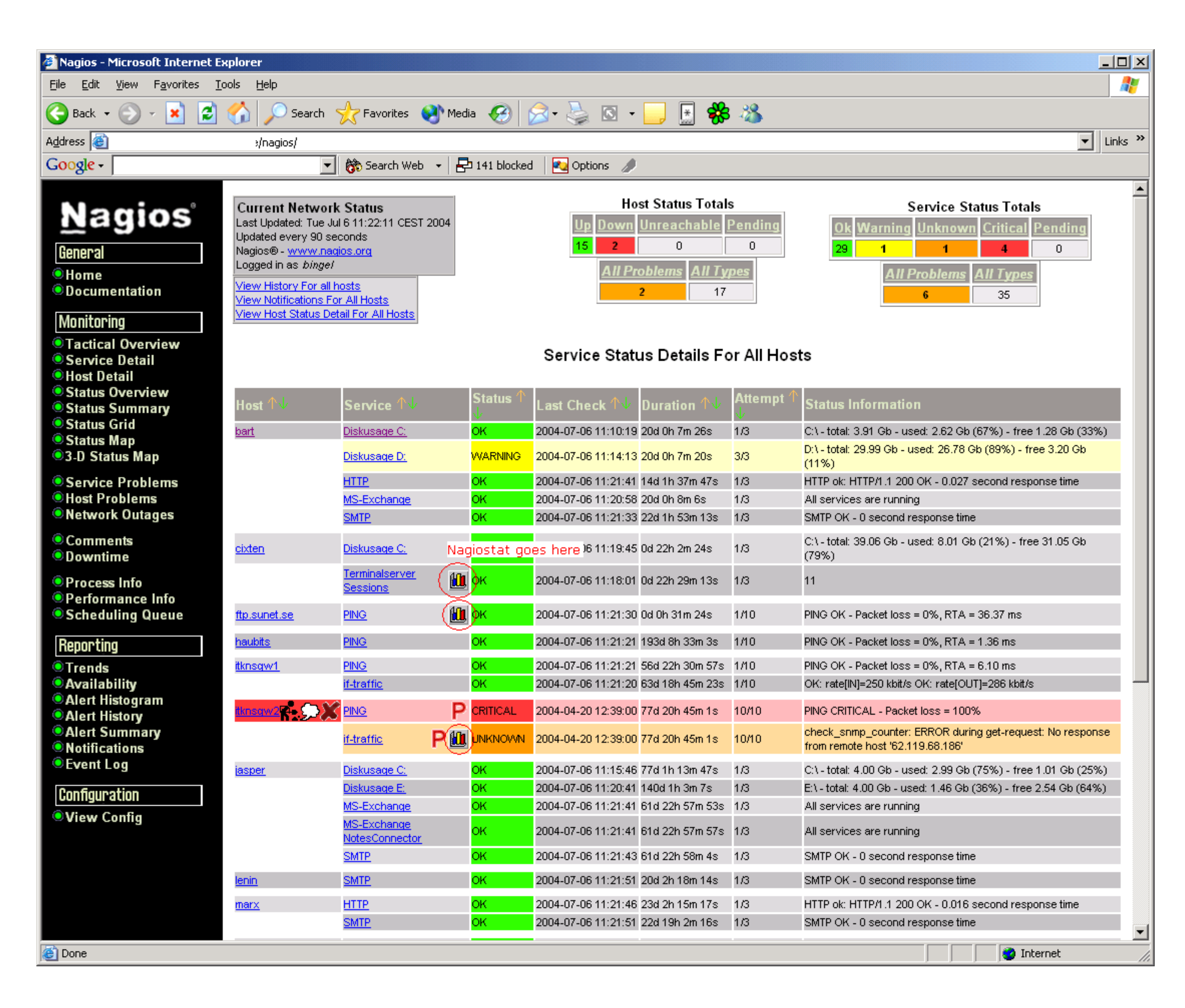


Legacy monitoring systems

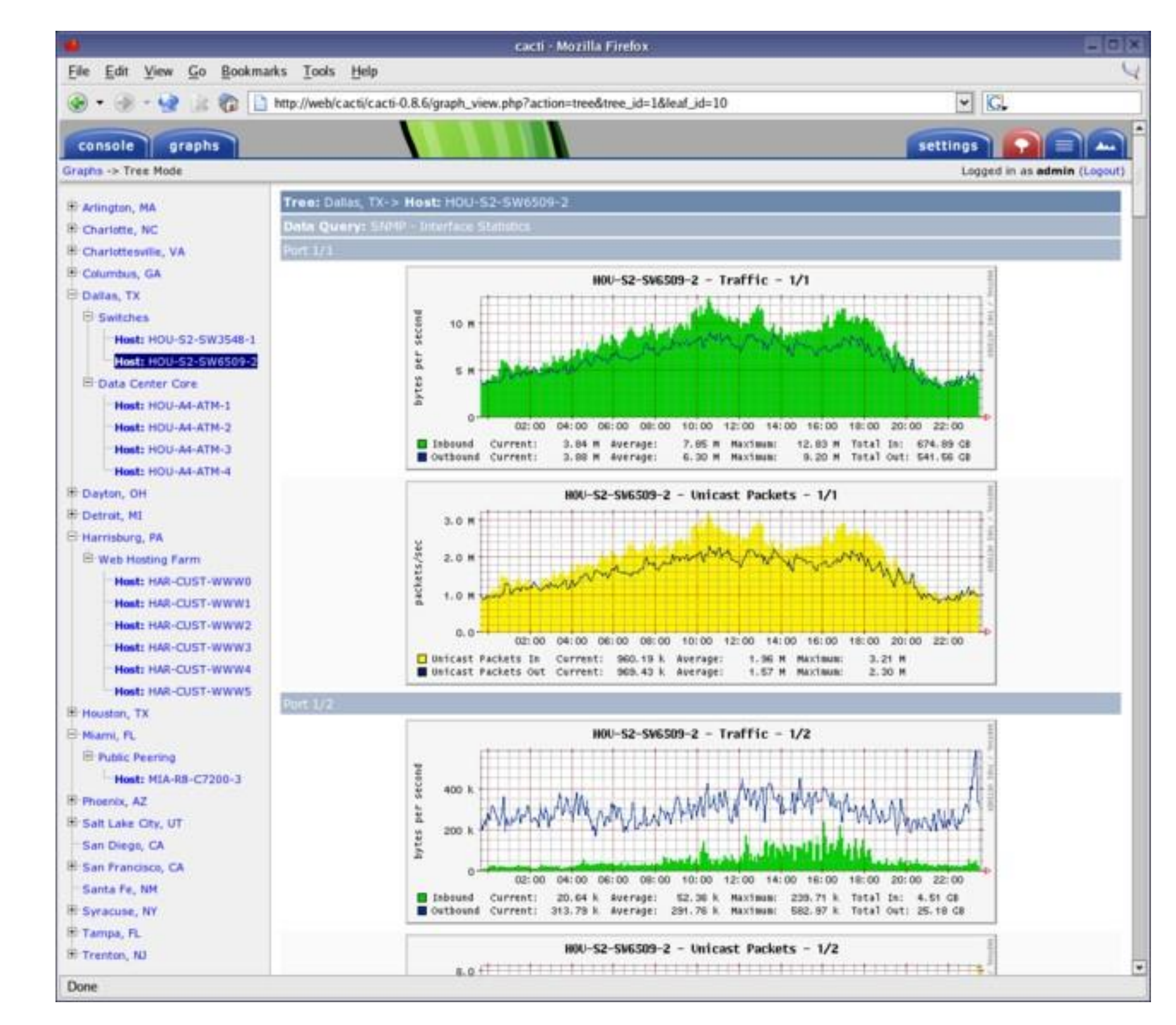


Nagios

Legacy monitoring systems

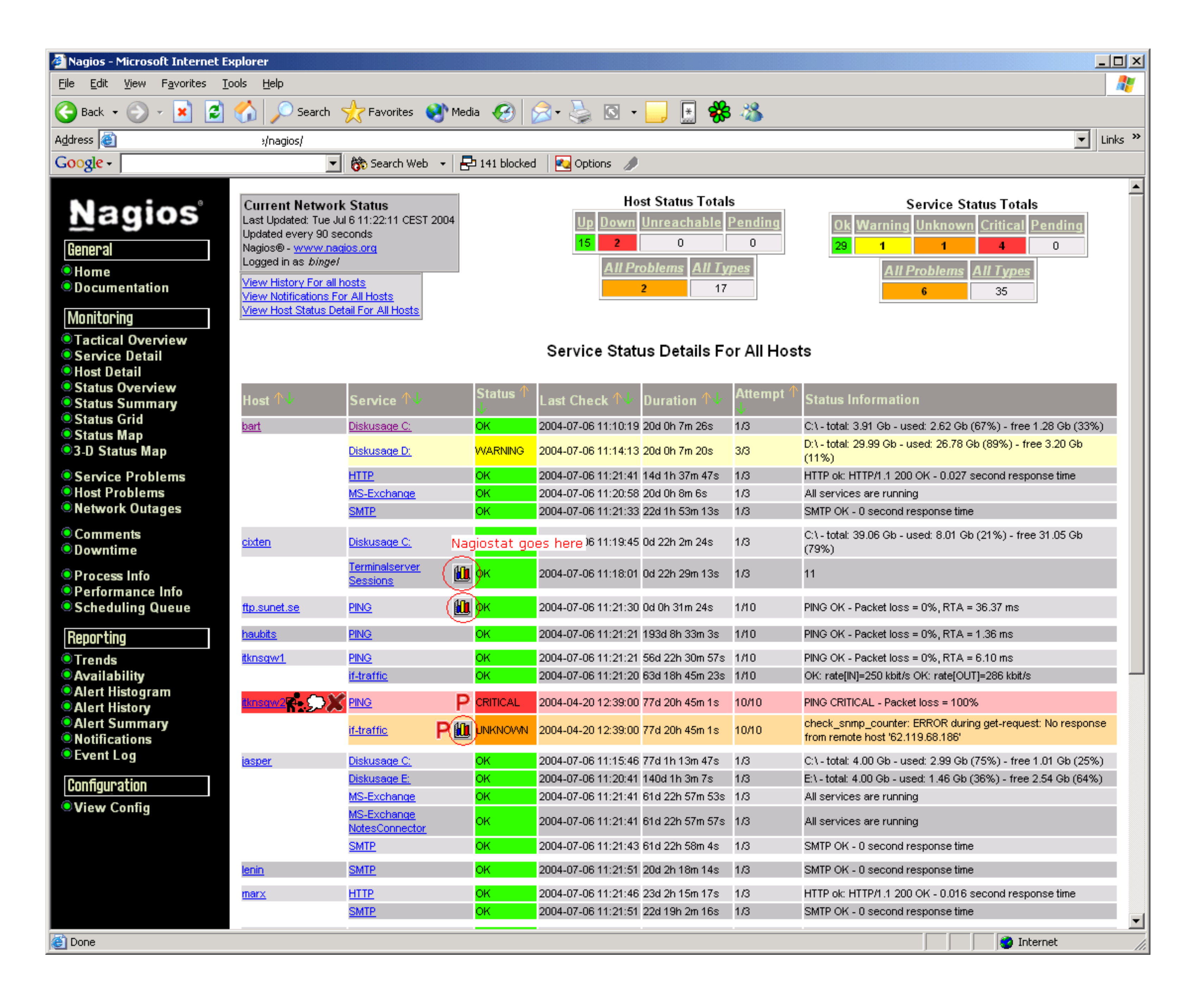


Nagios



Cacti

Legacy monitoring systems



Nagios

Cacti

Zabbix

Modern monitoring systems

NETDATA

netdata is a **monitoring agent**: you install it on all your systems:
- supports **auto-detection** and **zero configuration** for most applications and systems
- is **real-time**: every metric is on your dashboard in less than 1-second (collection to visualization)
- is **fast**: for a few thousand metrics per second, it needs < 1% CPU of a single core
- and **efficient**: it needs a few MB of RAM and no disk I/O at all while it runs
- also, it is **embeddable**, **extensible**, and **open-source** (GPL v3+)

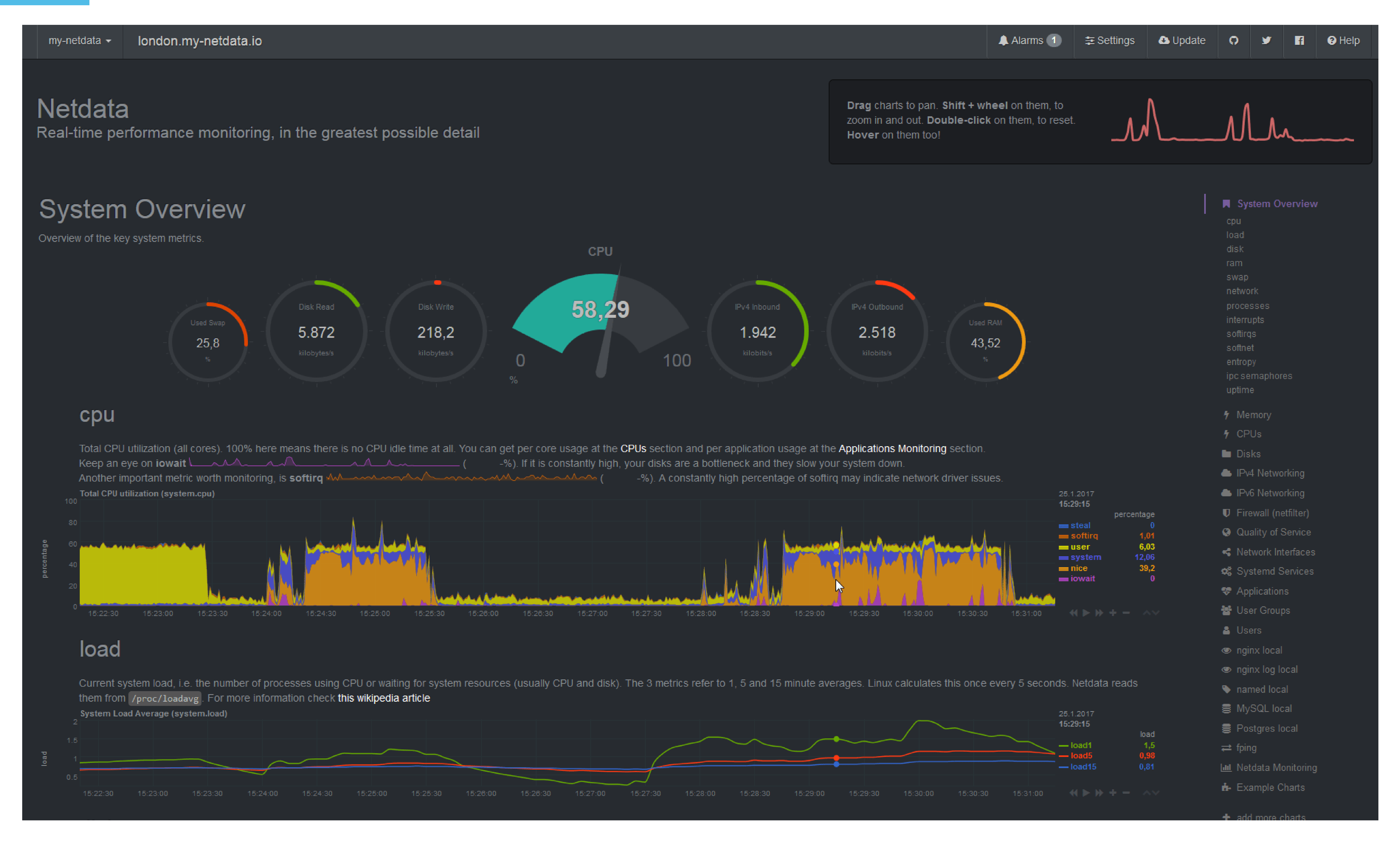
...and netdata runs everywhere:

netdata
simple. effective. awesome!
https://my-netdata.io

NETDATA

Netdata — Prometheus Exporter

Modern monitoring systems



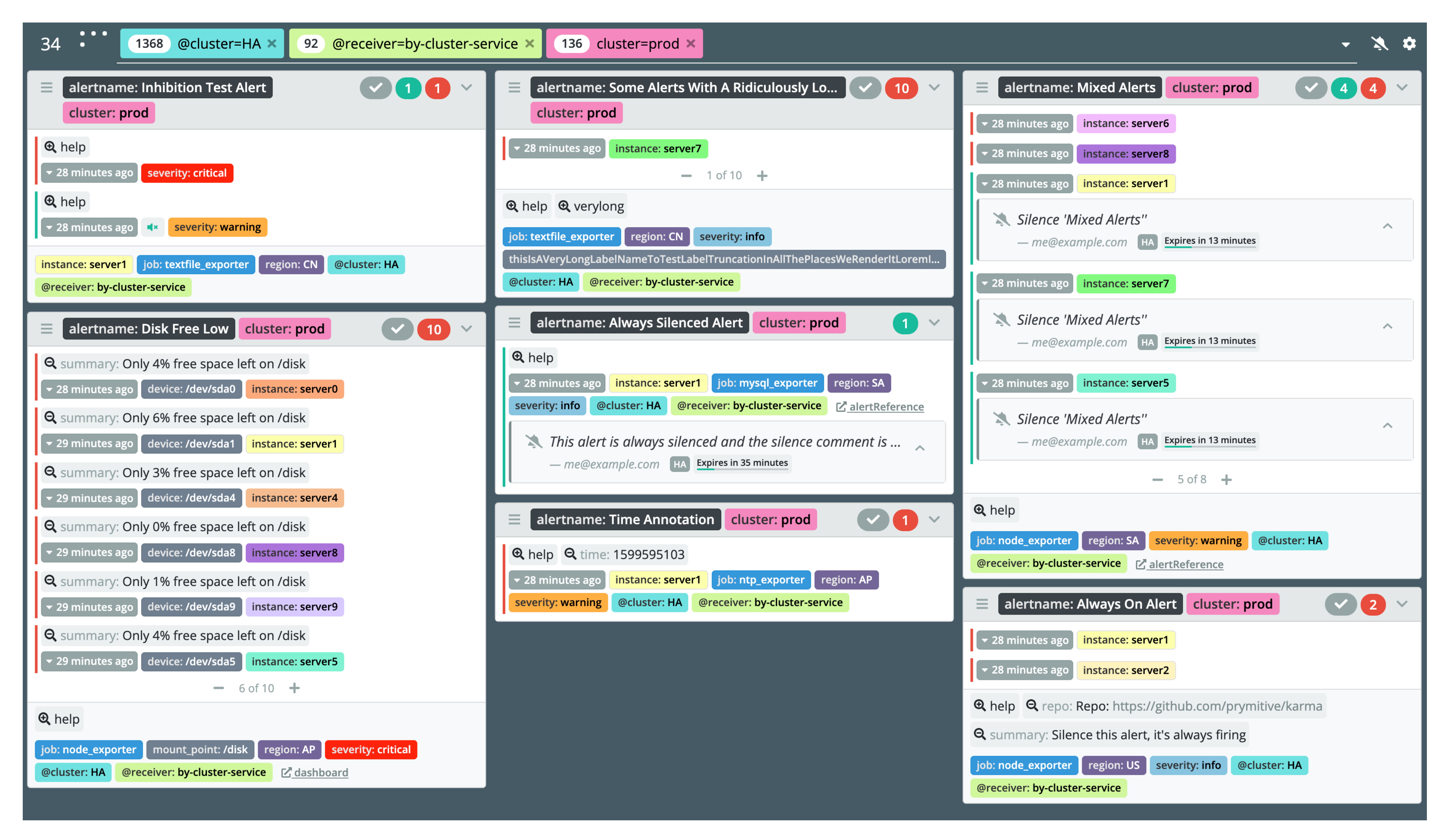
Netdata

Modern monitoring systems



Grafana

Modern monitoring systems



Karma

Legacy monitoring systems

МАЛЬЧИК, КОТОРЫЙ КРИЧАЛ «ВОЛКИ!»









Now we are here

- ~~The way from Legacy monitoring systems to Modern~~
- Monitoring Problems
- Legacy Infrastructure Service Discovery
- Prometheus Operator as the Solution
- PaaS Alerting and Dashboards — Monitoring As A Code
- Upgrades and Incidents
- Conclusion

Monitoring Problems

- Ops is the bottleneck

Monitoring Problems

- Ops is the bottleneck
- Time to market of business metrics

Monitoring Problems

- Ops is the bottleneck
- Time to market of business metrics
- Metrics, Dashboards, Alerts As A Code

Monitoring Problems

- Ops is the bottleneck
- Time to market of business metrics
- Metrics, Dashboards, Alerts As A Code
- More then one monitoring system

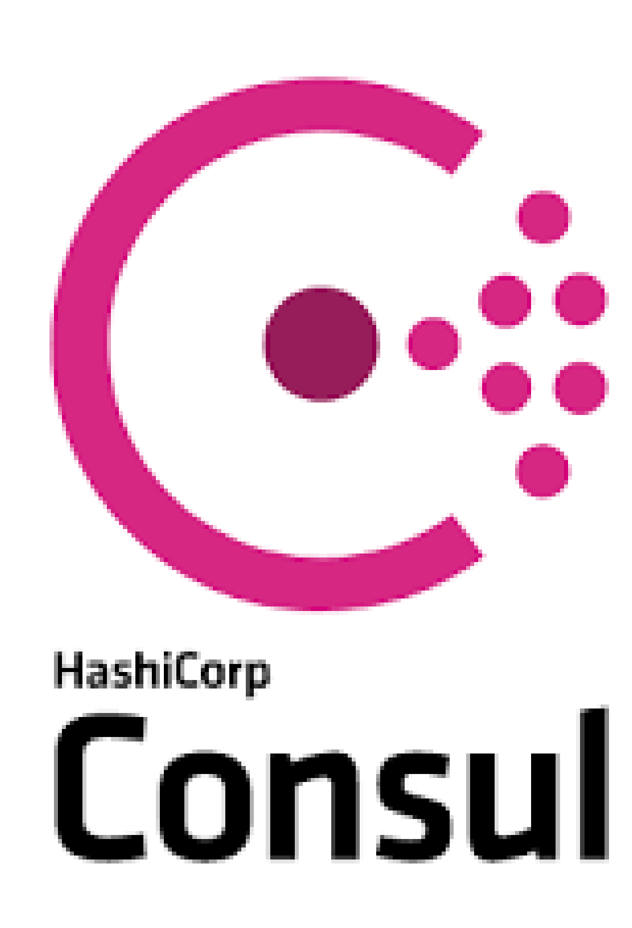
Now we are here

- ~~The way from Legacy monitoring systems to Modern~~
- ~~Monitoring Problems~~
- Legacy Infrastructure Service Discovery
- Prometheus Operator as the Solution
- PaaS Alerting and Dashboards — Monitoring As A Code
- Upgrades and Incidents
- Conclusion

Prometheus Service Discovery with Consul



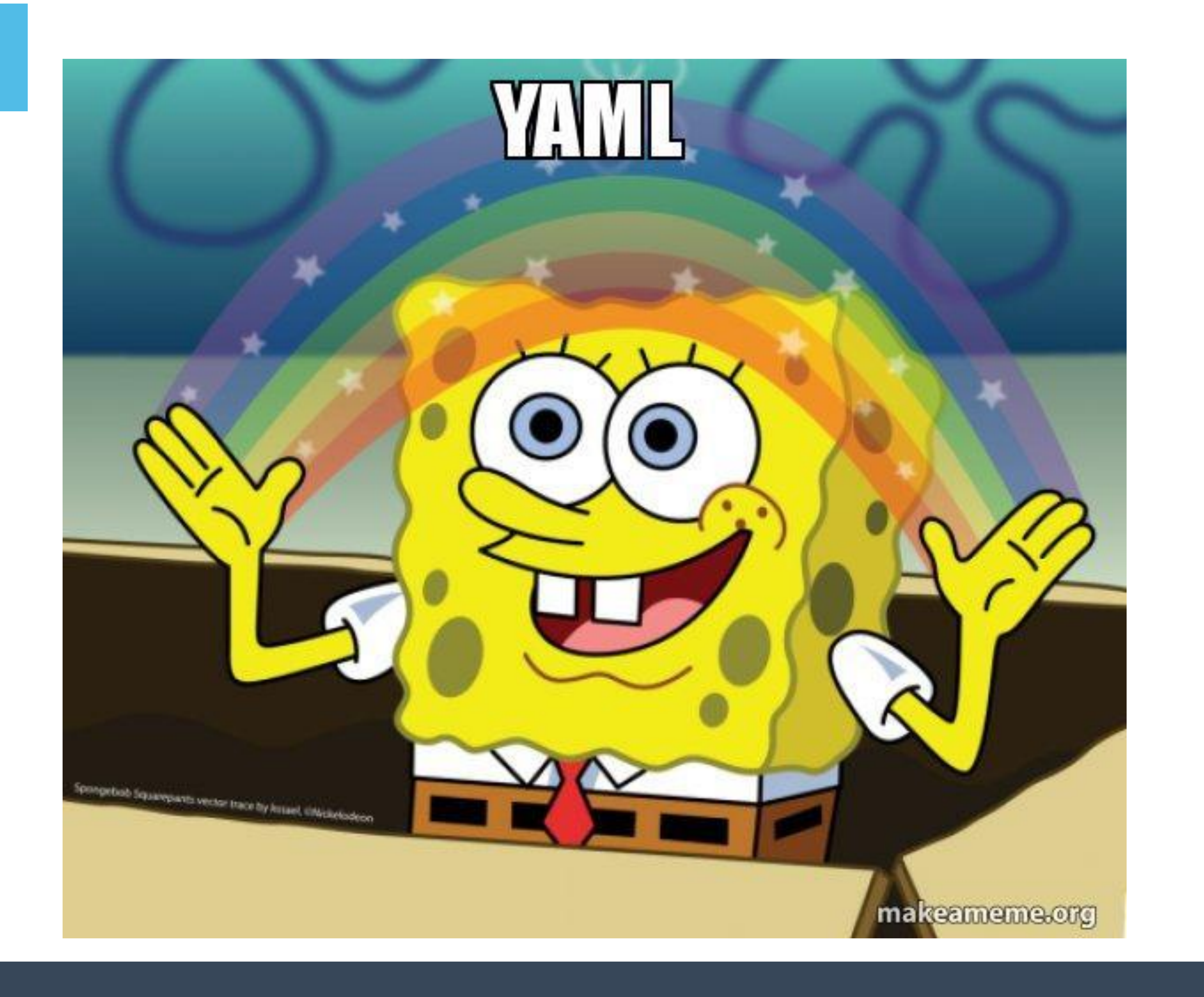
Metrics



Discovery



Agent


YAML
makeameme.org

Prometheus Service Discovery with Consul

```yaml
additionalScrapeConfigs:
  - job_name: consul
    consul_sd_configs:
      - server: 'consul-server.consul:8500'
        datacenter: {{ consul_dc }}
        tags: ["monitoring"]
    relabel_configs:
      - source_labels: [__meta_consul_service]
        target_label: job
      - source_labels: [__meta_consul_node]
        target_label: instance

  - job_name: netdata
    metrics_path: '/api/v1/allmetrics'
    params:
      format: [prometheus]
    consul_sd_configs:
      - server: 'consul-server.consul:8500'
        datacenter: {{ consul_dc }}
        tags: ["netdata"]
    relabel_configs:
      - source_labels: [__meta_consul_service]
        target_label: job
      - source_labels: [__meta_consul_node]
        target_label: instance
```

Now we are here

- ~~The way from Legacy monitoring systems to Modern~~
- ~~Monitoring Problems~~
- ~~Legacy Infrastructure Service Discovery~~
- Prometheus Operator as the Solution
- PaaS Alerting and Dashboards — Monitoring As A Code
- Upgrades and Incidents
- Conclusion

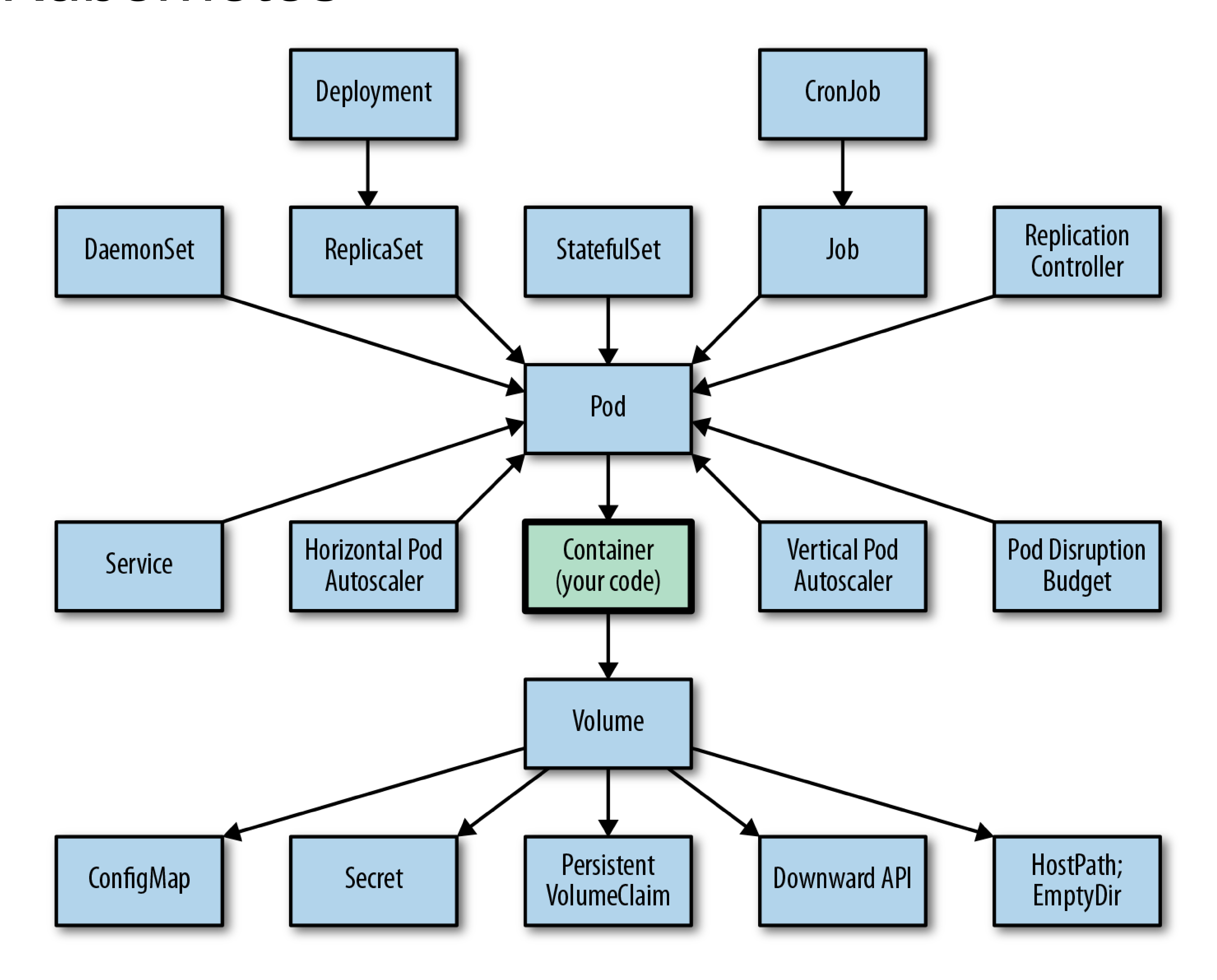
Kubernetes




WHAT GIVES PEOPLE
FEELINGS OF POWER
MONEY
STATUS
Run your own kubernetes cluster



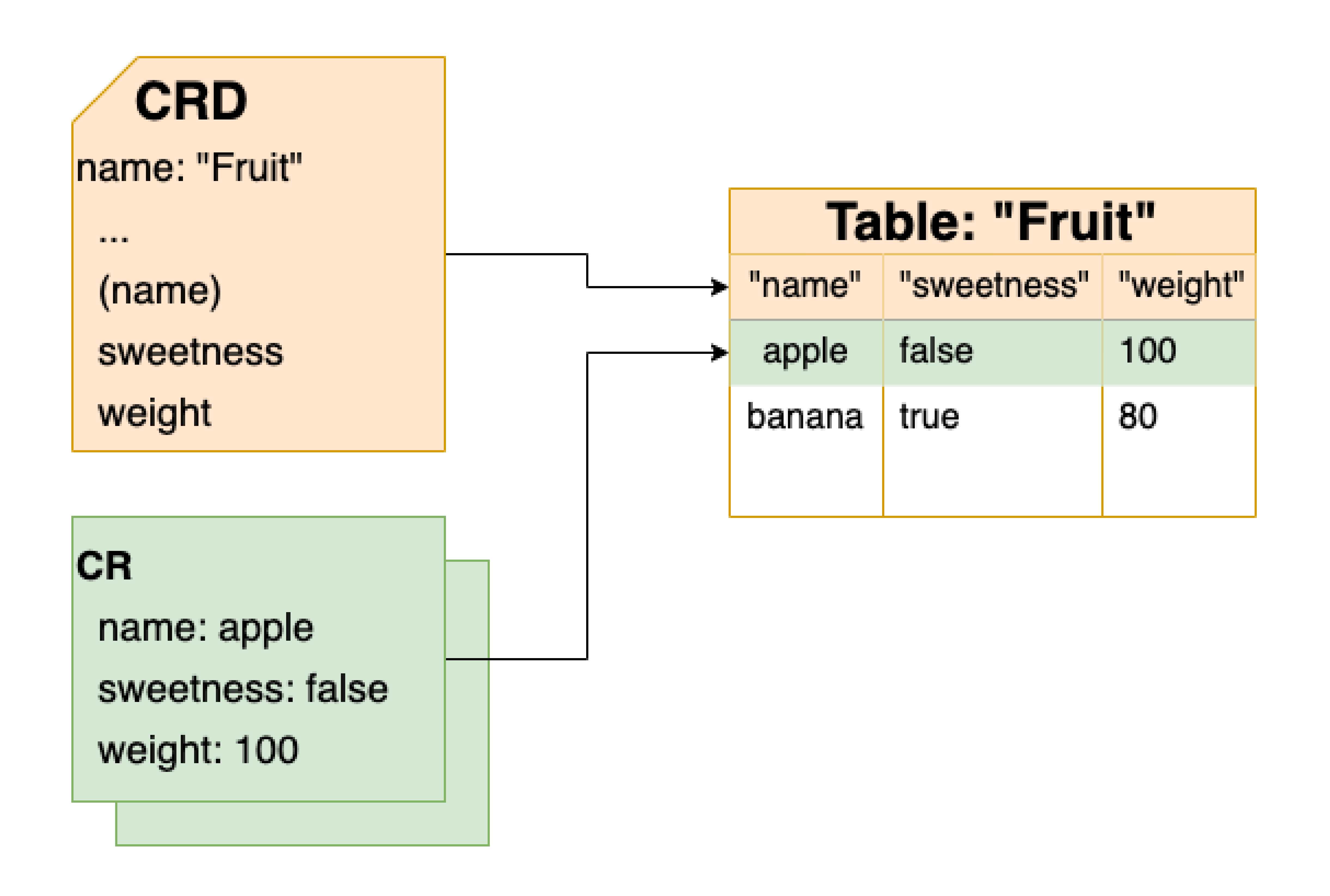
Kubernetes


Deployment
CronJob
DaemonSet
ReplicaSet
StatefulSet
Job
Replication Controller
Pod
Service
Horizontal Pod Autoscaler
Container (your code)
Vertical Pod Autoscaler
Pod Disruption Budget
Volume
ConfigMap
Secret
Persistent VolumeClaim
Downward API
HostPath; EmptyDir

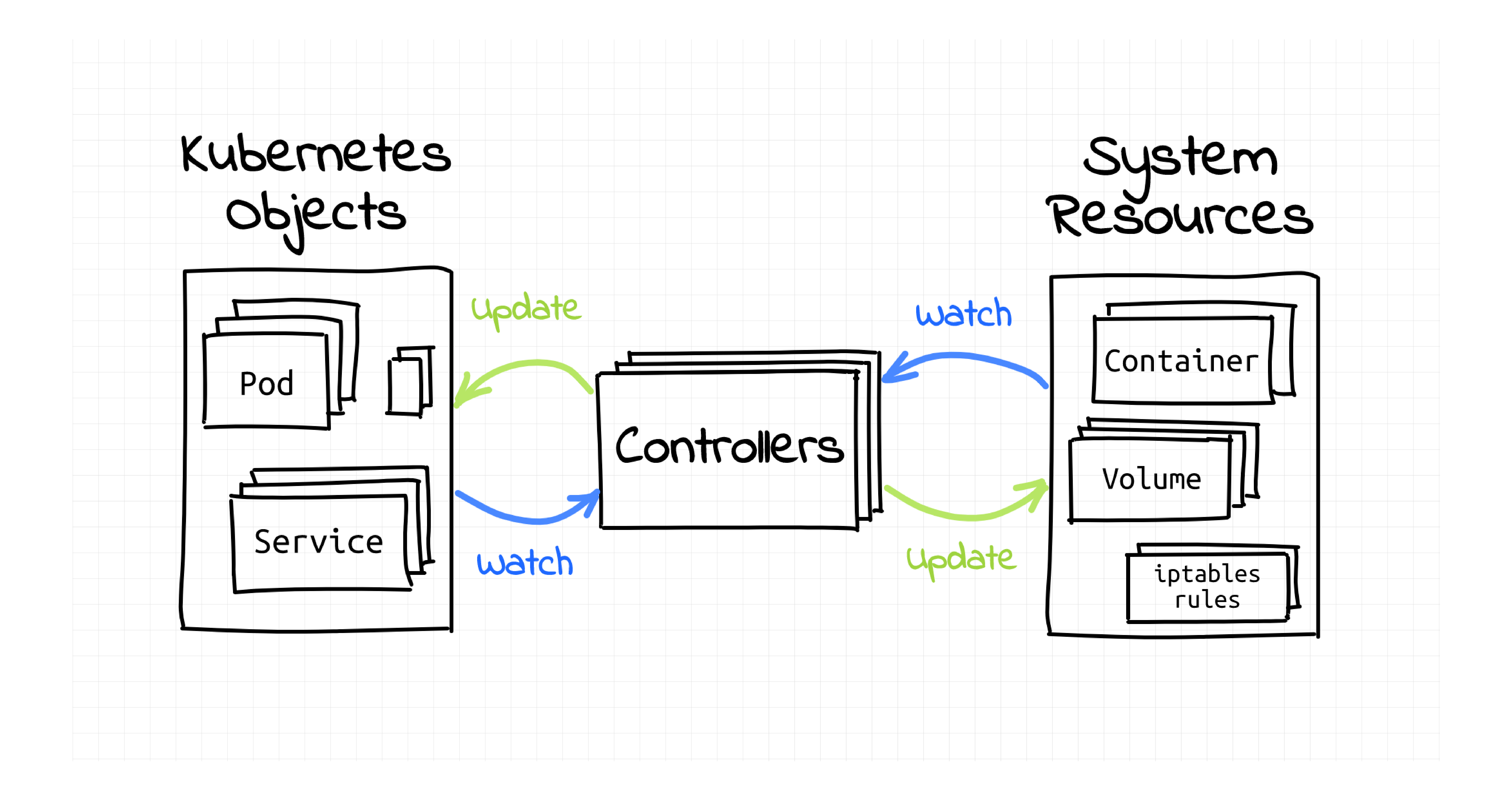
Kubernetes Custom Resource Definition



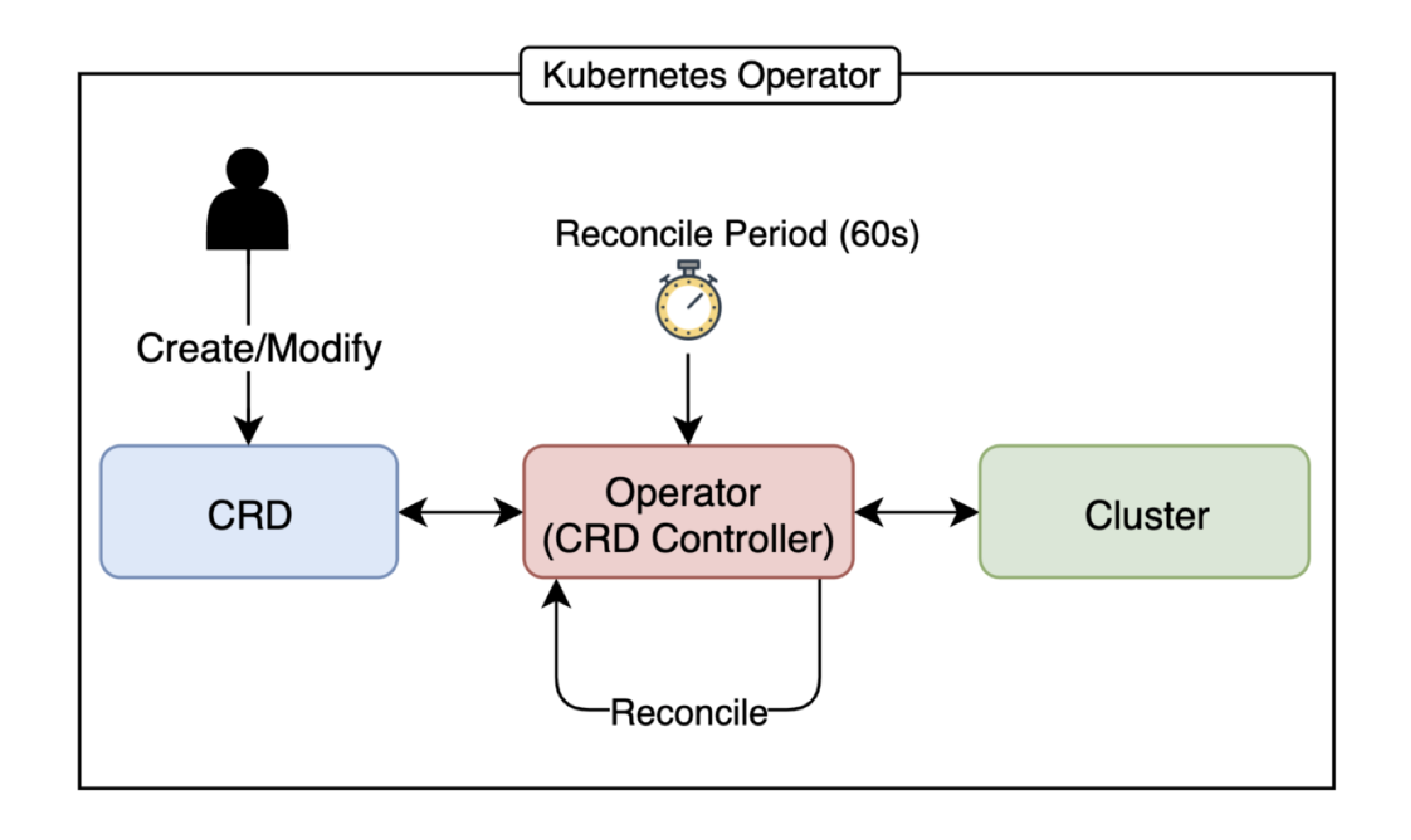
Kubernetes Custorm Resource Definition



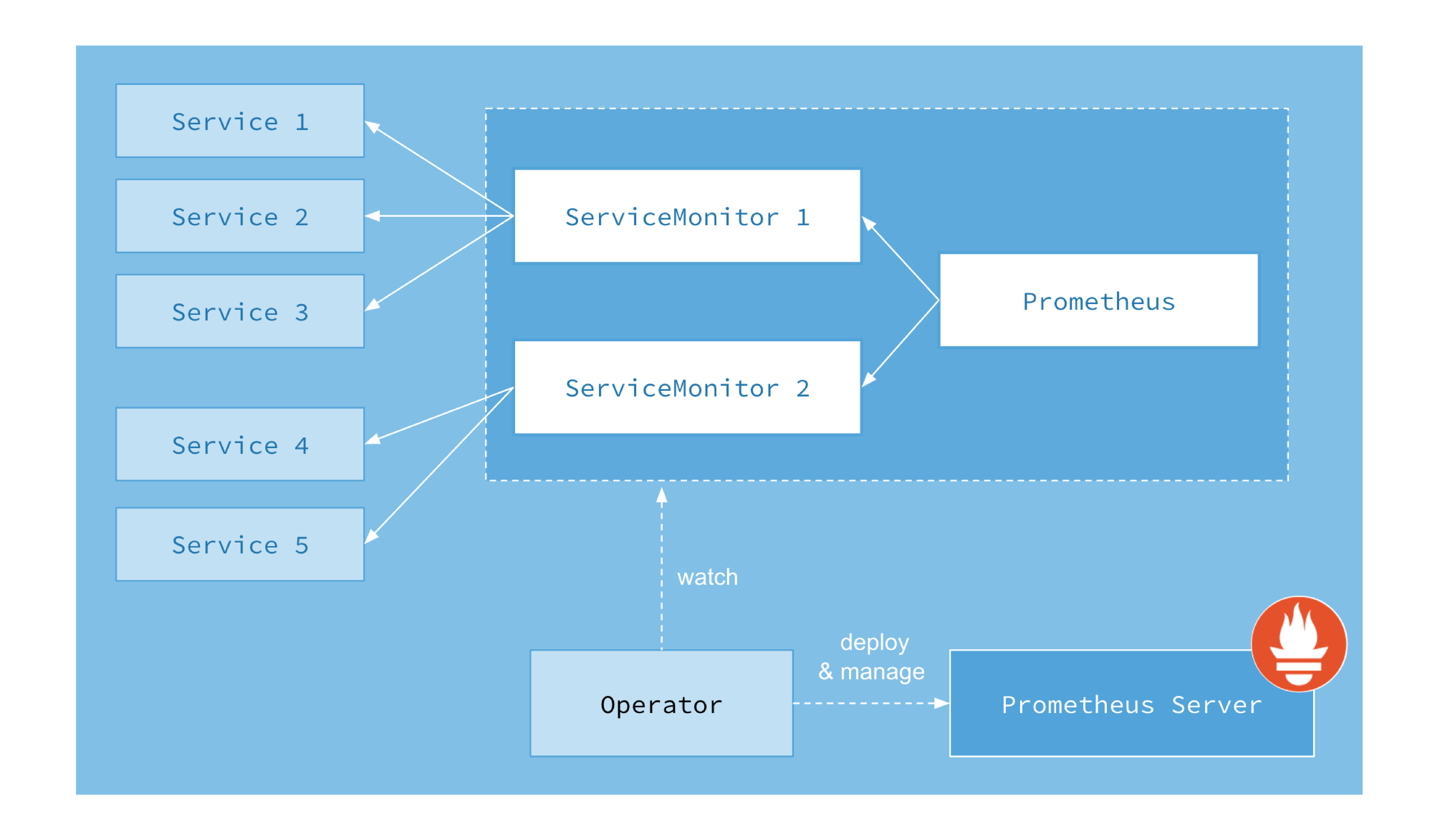
Kubernetes Operator



Kubernetes Operator



Prometheus Operator — Architecture Overview



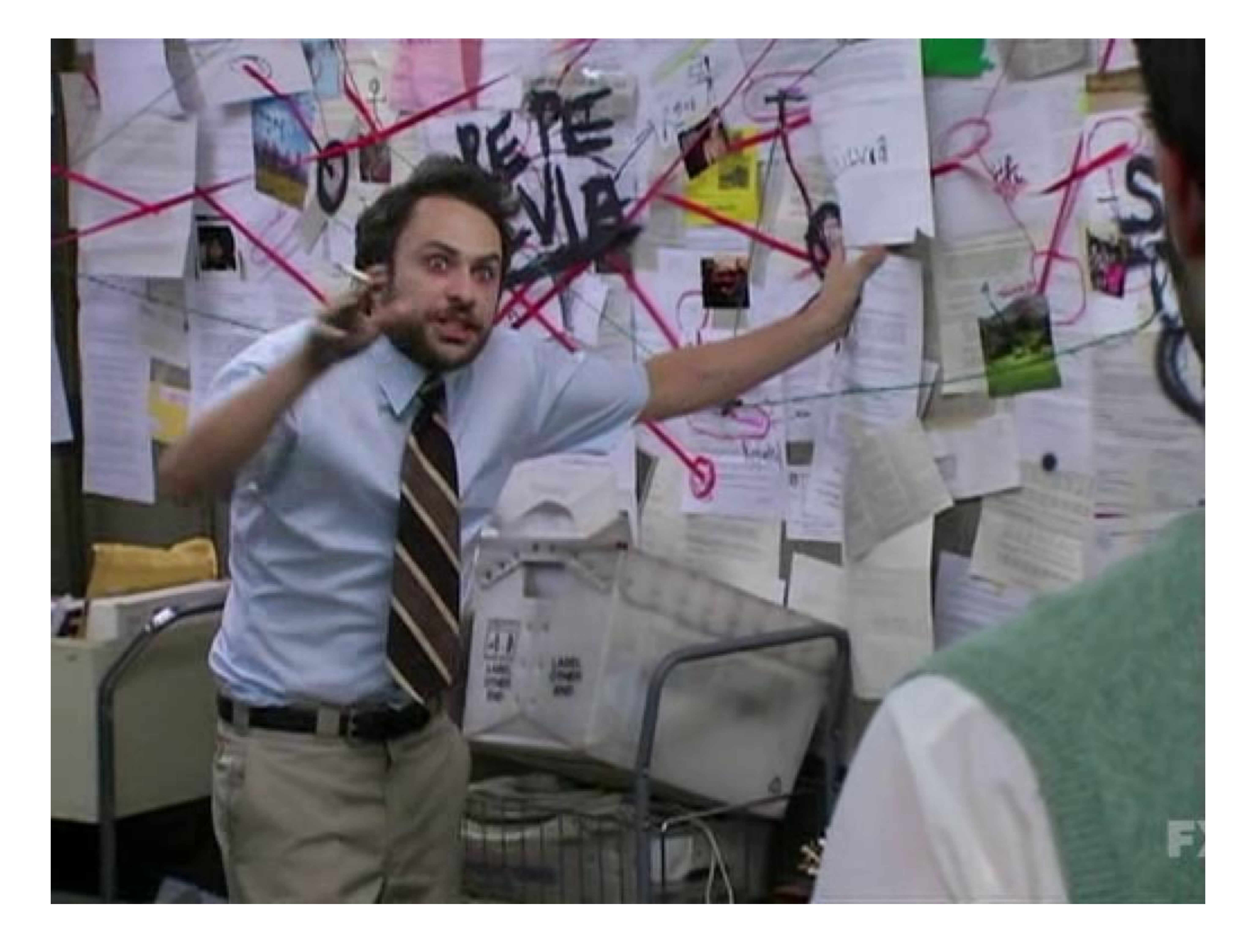
ServiceMonitor

```yaml
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    app: prometheus-operator-alertmanager
    chart: prometheus-operator-9.3.2
    heritage: Helm
    release: prometheus-operator
  name: prometheus-operator-alertmanager
  namespace: monitoring
spec:
  endpoints:
  - path: /metrics
    port: web
  namespaceSelector:
    matchNames:
    - monitoring
  selector:
    matchLabels:
      app: prometheus-operator-alertmanager
      release: prometheus-operator
```

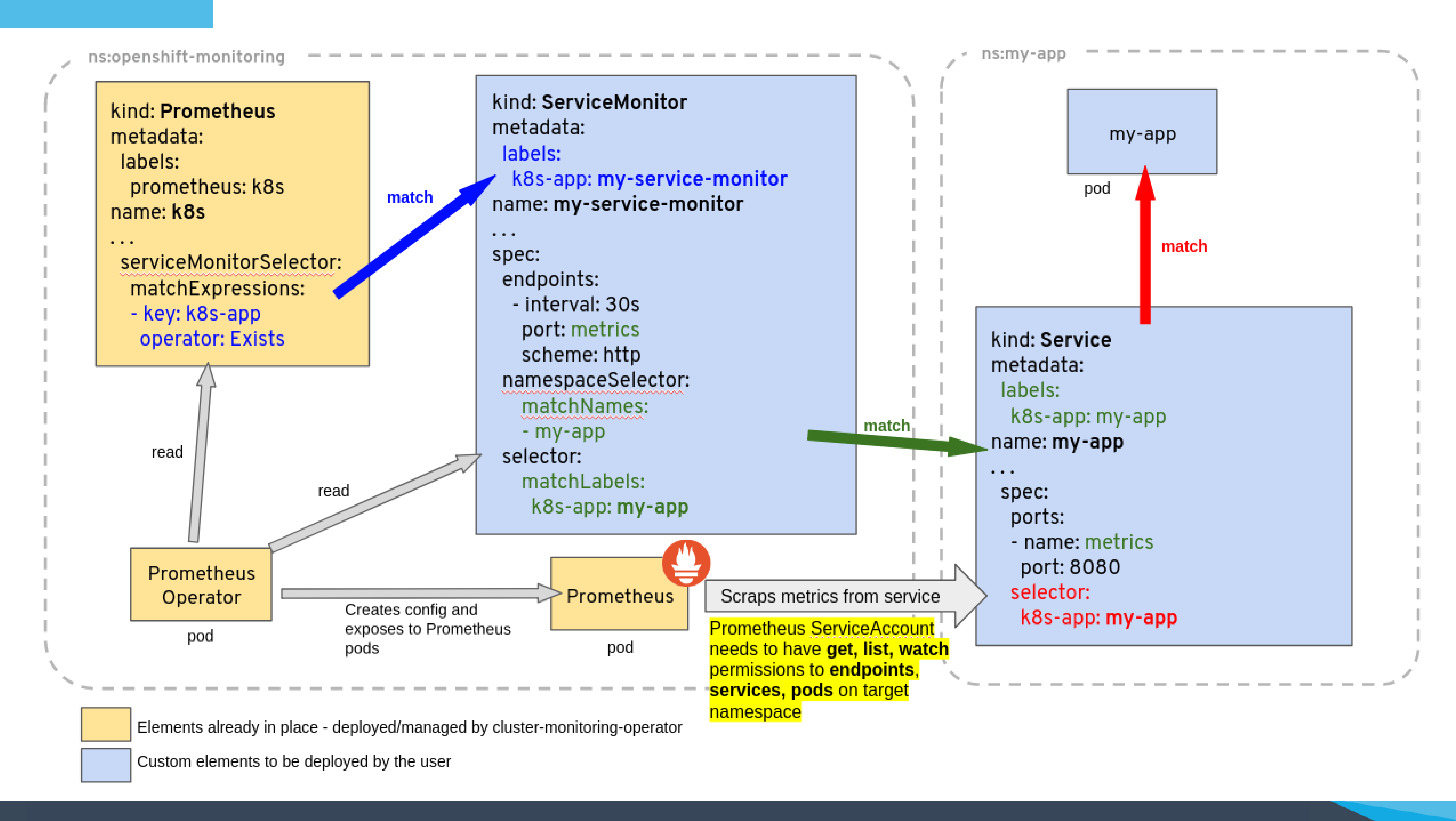
ServiceMonitor

```
~ ⎈ wsp.production  monitoring  %  k get servicemonitors.monitoring.coreos.com
NAME                                          AGE
prometheus-operator-alertmanager              199d
prometheus-operator-apiserver                 199d
prometheus-operator-coredns                   199d
prometheus-operator-grafana                   199d
prometheus-operator-kube-controller-manager   199d
prometheus-operator-kube-etcd                 199d
prometheus-operator-kube-proxy                187d
prometheus-operator-kube-scheduler            199d
prometheus-operator-kube-state-metrics        199d
prometheus-operator-kubelet                   199d
prometheus-operator-node-exporter             199d
prometheus-operator-operator                  199d
prometheus-operator-prometheus                199d
```

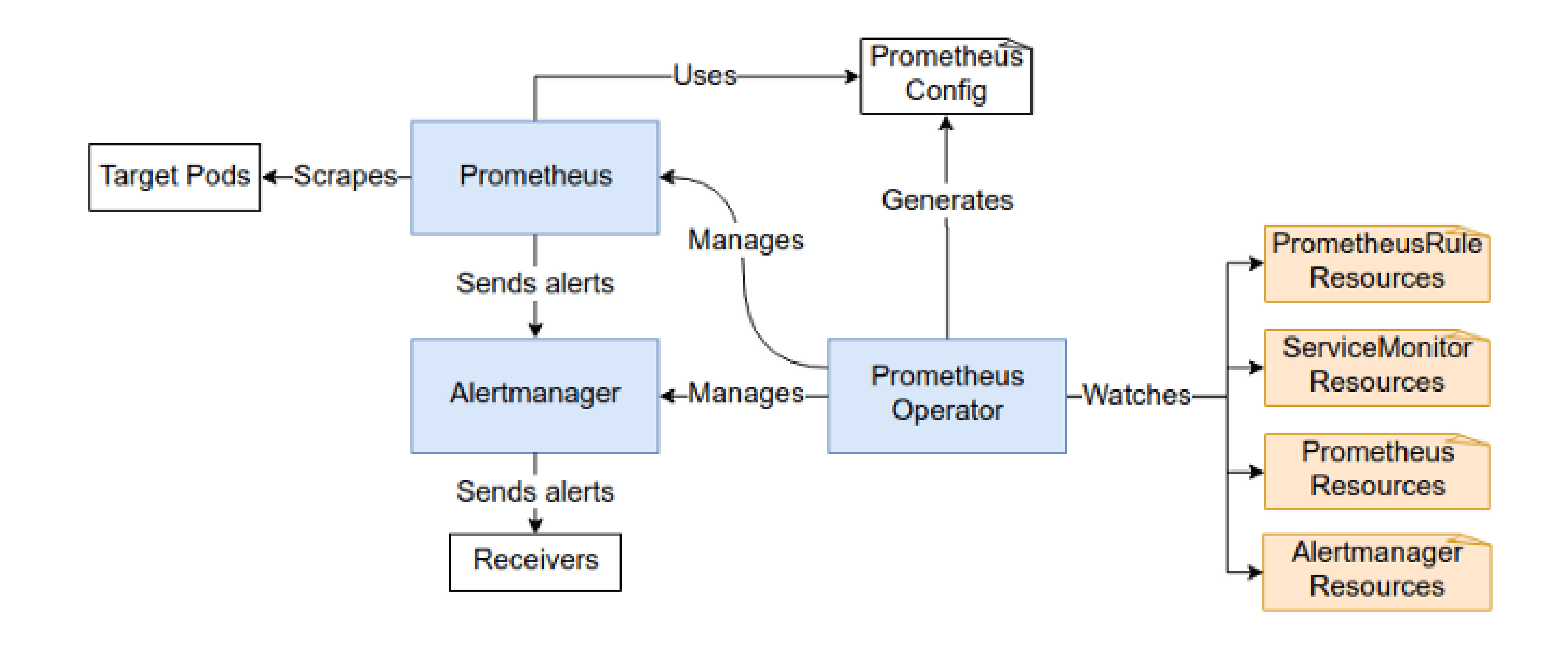
ServiceMonitor



ServiceMonitor



Prometheus Operator — how it works



PrometheusRules

```yaml
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  annotations:
    prometheus-operator-validated: "true"
  labels:
    app: prometheus-operator
    chart: prometheus-operator-9.3.2
    heritage: Helm
    release: prometheus-operator
  name: prometheus-operator-etcd
  namespace: monitoring
spec:
  groups:
  - name: etcd
    rules:
    - alert: etcdInsufficientMembers
      annotations:
        message: 'etcd cluster "{{ $labels.job }}": insufficient members ({{ $value
          }}).'
      expr: sum(up{job=~".*etcd.*"} == bool 1) by (job) < ((count(up{job=~".*etcd.*"})
        by (job) + 1) / 2)
      for: 3m
      labels:
        severity: critical
```

PrometheusRules — alerts out-of-the-box

```
~ ⎈ wsp.production monitoring % k get prometheusrules.monitoring.coreos.com
NAME                                                   AGE
prometheus-operator-alertmanager                       199d
prometheus-operator-blackbox                           151d
prometheus-operator-cert-manager                       151d
prometheus-operator-etcd                               199d
prometheus-operator-general                            199d
prometheus-operator-k8s                                199d
prometheus-operator-kube-apiserver-error               117d
prometheus-operator-kube-apiserver                     199d
prometheus-operator-kube-prometheus-node-recording     199d
prometheus-operator-kube-scheduler                     199d
prometheus-operator-kubernetes-absent                  117d
prometheus-operator-kubernetes-apps                    199d
prometheus-operator-kubernetes-resources               199d
prometheus-operator-kubernetes-storage                 199d
prometheus-operator-kubernetes-system                  199d
prometheus-operator-kubernetes-system-apiserver        199d
prometheus-operator-kubernetes-system-controller-manager 199d
prometheus-operator-kubernetes-system-kubelet          199d
prometheus-operator-kubernetes-system-scheduler        199d
```

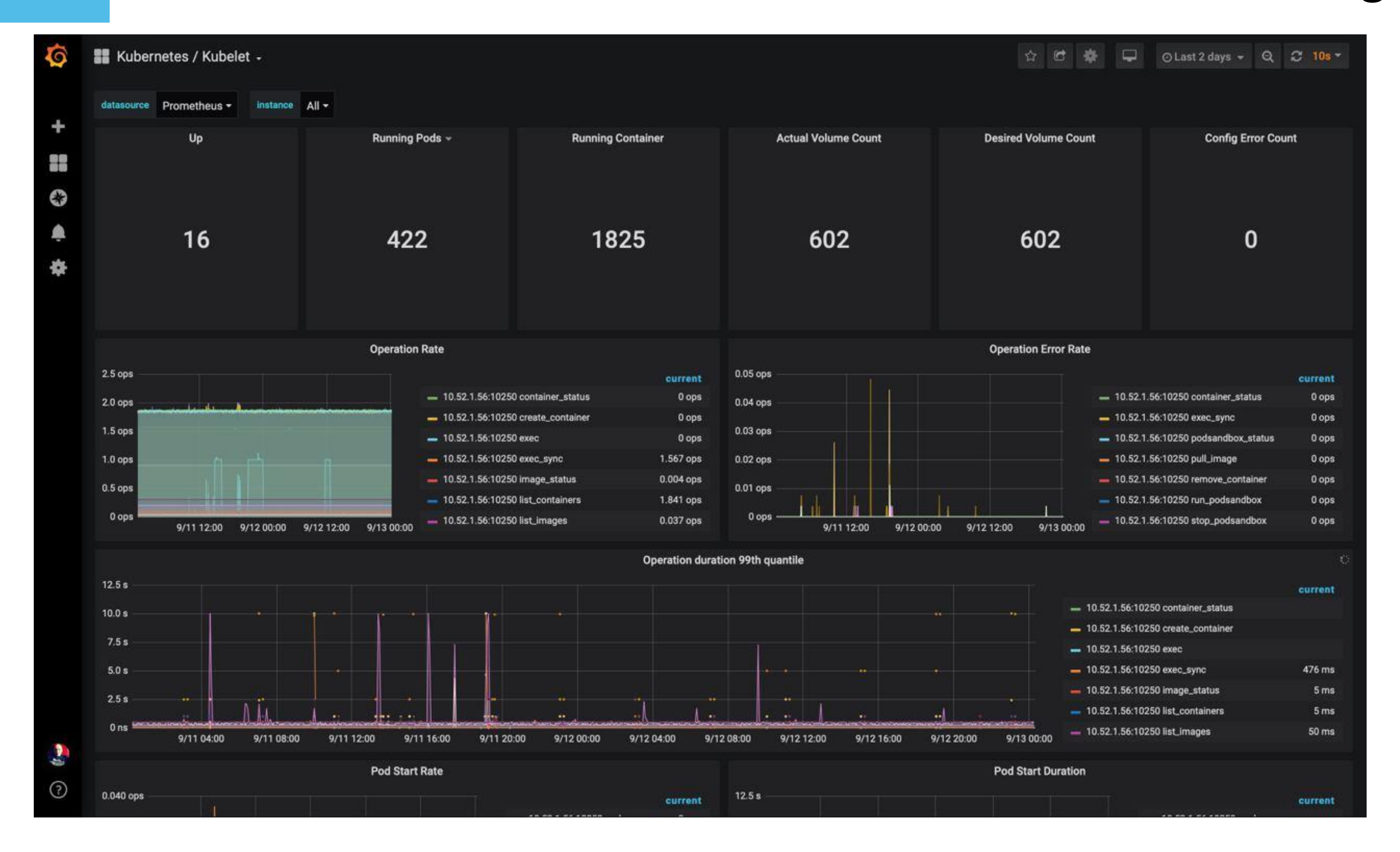
Kubernetes-mixin — all-in-one k8s monitoring

```
~ ⎈ wsp.production  monitoring  %  k get configmaps | grep prometheus-operator
prometheus-operator-apiserver                           1      201d
prometheus-operator-cluster-total                       1      201d
prometheus-operator-controller-manager                  1      201d
prometheus-operator-etcd                                1      201d
prometheus-operator-grafana                             2      201d
prometheus-operator-grafana-config-dashboards           1      201d
prometheus-operator-grafana-datasource                  1      201d
prometheus-operator-grafana-test                        1      201d
prometheus-operator-k8s-coredns                         1      201d
prometheus-operator-k8s-resources-cluster               1      201d
prometheus-operator-k8s-resources-namespace             1      201d
prometheus-operator-k8s-resources-node                  1      201d
prometheus-operator-k8s-resources-pod                   1      201d
prometheus-operator-k8s-resources-workload              1      201d
prometheus-operator-k8s-resources-workloads-namespace   1      201d
prometheus-operator-kubelet                             1      201d
prometheus-operator-namespace-by-pod                    1      201d
prometheus-operator-namespace-by-workload               1      201d
prometheus-operator-node-cluster-rsrc-use               1      201d
prometheus-operator-node-rsrc-use                       1      201d
prometheus-operator-nodes                               1      201d
prometheus-operator-persistentvolumesusage              1      201d
prometheus-operator-pod-total                           1      201d
prometheus-operator-pods                                1      120d
prometheus-operator-prometheus                          1      201d
prometheus-operator-proxy                               1      189d
prometheus-operator-scheduler                           1      201d
prometheus-operator-statefulset                         1      201d
prometheus-operator-workload-total                      1      201d
prometheus-prometheus-operator-prometheus-rulefiles-0   32     40d
```

Kubernetes-mixin — all-in-one k8s monitoring



Kubernetes-mixin — all-in-one k8s monitoring



Longterm storage Overview





Longterm storage Overview





Now we are here

- ~~The way from Legacy monitoring systems to Modern~~
- ~~Monitoring Problems~~
- ~~Legacy Infrastructure Service Discovery~~
- ~~Prometheus Operator as the Solution~~
- PaaS Alerting and Dashboards — Monitoring As A Code
- Upgrades and Incidents
- Conclusion

Deploy Helm Chart via Ansible



PROMETHEUS-OPERATOR
VS
KUBE-PROMETHEUS-STACK



Further development has moved to prometheus-community/helm-charts. The chart has been renamed kube-prometheus-stack to more clearly reflect that it installs the `kube-prometheus` project stack, within which Prometheus Operator is only one component.

PaaS Alerting and Dashboards — Monitoring As A Code







PaaS Alerting and Dashboards — Monitoring As A Code

```yaml
- name: Applying prometheus crds
  k8s:
    definition:
      apiVersion: monitoring.coreos.com/v1
      kind: PrometheusRule
      metadata:
        labels:
          app: prometheus-operator
          generated: 'true'
          release: prometheus-operator
        name: "prometheus-operator-{{ (item.path | basename | splitext)[0] }}"
        namespace: monitoring
      spec:
        groups:
          - name: "{{ item.path | basename }}"
            rules: "{{ lookup('file', item.path) | from_yaml }}"
  loop: "{{ alerting_sources.files }}"
  loop_control:
    label: "{{ item.path }}"
```

PaaS Alerting and Dashboards — Monitoring As A Code

```yaml
- alert: netdataMetricsUnreachable
  expr: netdata:reachability:bool == 0
  for: 5m
  labels:
    severity: critical
    routing_key: ops
  annotations:
    message: "{{ $labels.instance }} netdata metrics unreachable from prometheus for 5m."
    runbook: troubleshooting/node.md#netdataMetricsUnreachable
    dashboard_uid: 6bab49de
    panel_metric: netdata:reachability:bool{{"{"}}instance="{{ $labels.instance }}"{{"}"}}
```

PaaS Alerting and Dashboards — Monitoring As A Code

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  labels:
    grafana_dashboard: "true"
    grafana_dashboard_custom: "true"
  namespace: monitoring
  name: grafana-dashboard-{{ (item.path | basename | splitext)[0] }}
data:
  {{'grafana-dashboard-%s' | format( item.path | basename ) }}: >
  {{ lookup('file', item.path) | indent(6, first=True, blank=True) }}
```


{ }
JSON Statham

Now we are here

- ~~The way from Legacy monitoring systems to Modern~~
- ~~Monitoring Problems~~
- ~~Legacy Infrastructure Service Discovery~~
- ~~Prometheus Operator as the Solution~~
- ~~PaaS Alerting and Dashboards — Monitoring As A Code~~
- Upgrades and Incidents
- Conclusion

Upgrades and Incidents



Upgrades and Incidents

- ETCD endpoints — не работают из коробки

Upgrades and Incidents

- ETCD endpoints — не работают из коробки
- Изменение CRD при апгрейде = fail upgrade

Upgrades and Incidents

- ETCD endpoints — не работают из коробки
- Изменение CRD при апгрейде = fail upgrade
- Удаление ServiceMonitor при удаление helm-релиза

Upgrades and Incidents

- ETCD endpoints — не работают из коробки
- Изменение CRD при апгрейде = fail upgrade
- Удаление ServiceMonitor при удаление helm-релиза
- Неоптимальные параметры ServiceMonitor

Upgrades and Incidents

- ETCD endpoints — не работают из коробки
- Изменение CRD при апгрейде = fail upgrade
- Удаление ServiceMonitor при удаление helm-релиза
- Неоптимальные параметры ServiceMonitor
- Обновление конфигурации Prometheus — instance restart

Upgrades and Incidents

- ETCD endpoints — не работают из коробки
- Изменение CRD при апгрейде = fail upgrade
- Удаление ServiceMonitor при удаление helm-релиза
- Неоптимальные параметры ServiceMonitor
- Обновление конфигурации Prometheus — instance restart
- Отсутствие метрик (nodata) — тоже метрика

Upgrades and Incidents

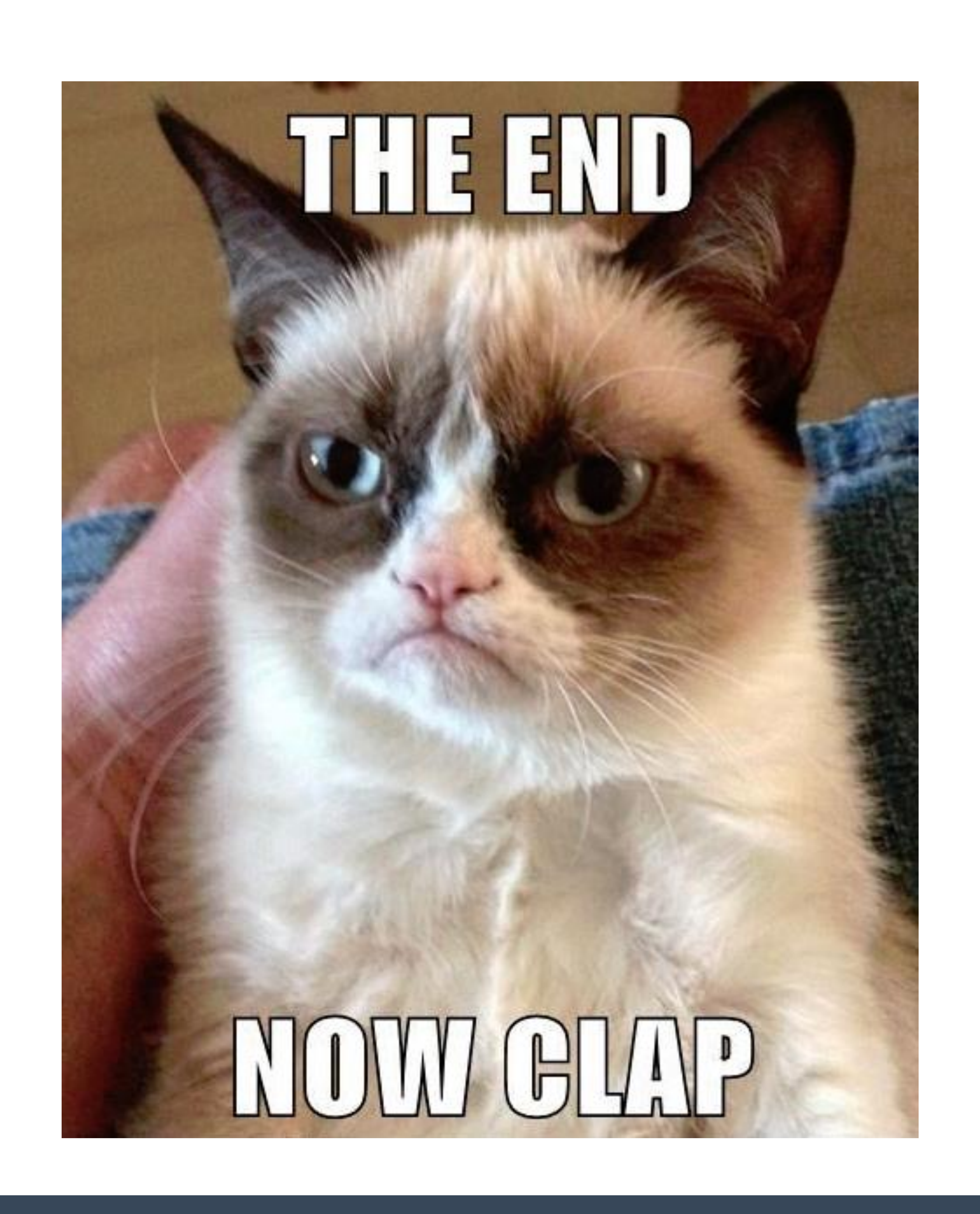
- ETCD endpoints — не работают из коробки
- Изменение CRD при апгрейде = fail upgrade
- Удаление ServiceMonitor при удаление helm-релиза
- Неоптимальные параметры ServiceMonitor
- Обновление конфигурации Prometheus — instance restart
- Отсутствие метрик (nodata) — тоже метрика
- Kube-apiserver умирает при 4GB при деплое helm-chart

Now we are here

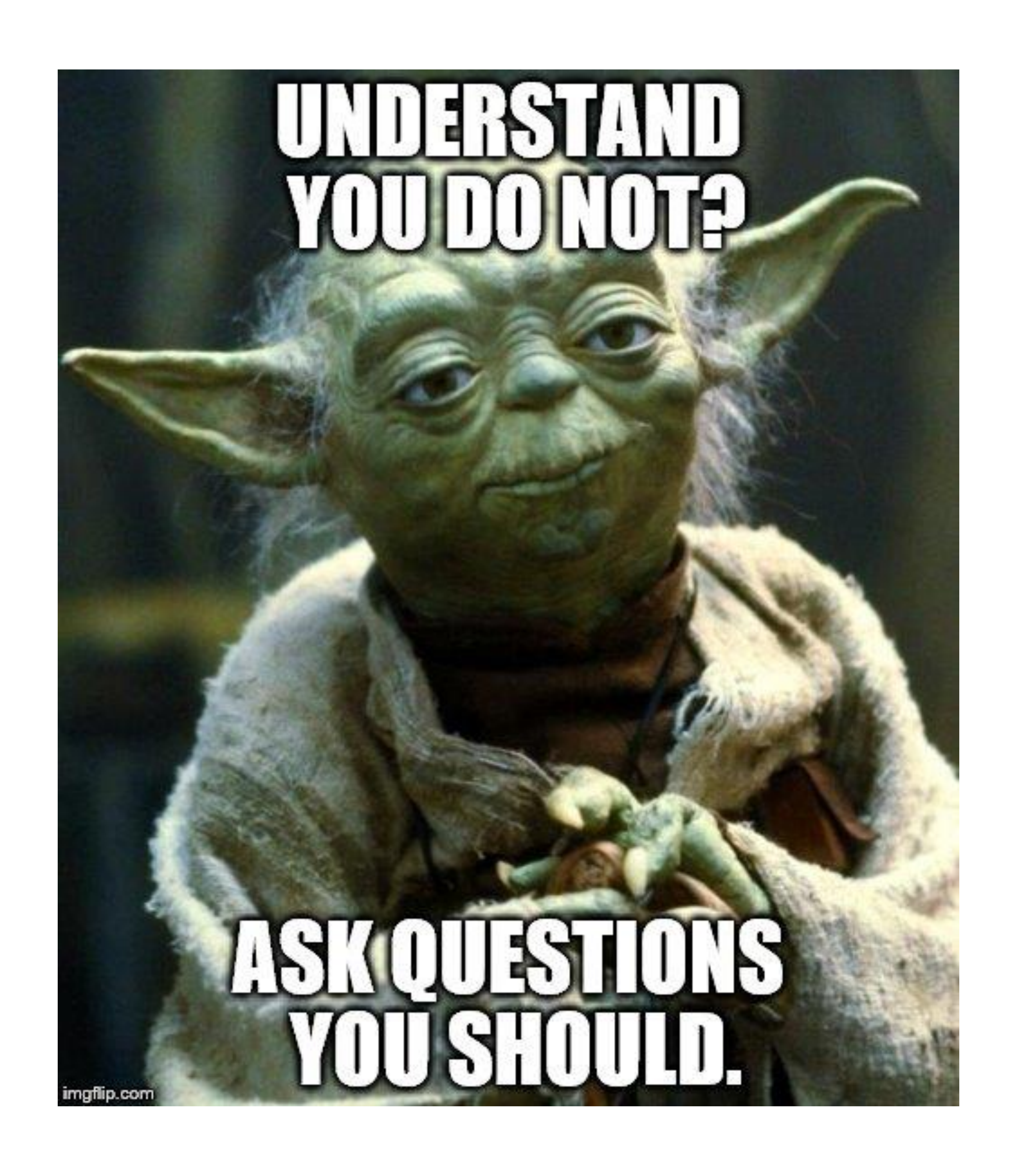
- ~~The way from Legacy monitoring systems to Modern~~
- ~~Monitoring Problems~~
- ~~Legacy Infrastructure Service Discovery~~
- ~~Prometheus Operator as the Solution~~
- ~~PaaS Alerting and Dashboards — Monitoring As A Code~~
- ~~Upgrades and Incidents~~
- Conclusion

Выводы

- “Админы” больше не bottleneck
- Разработчики самостоятельно доставляют бизнес-метрики
- Решение легко встраивается в любой k8s pipeline
- Легко ложится на infrastructure as a code


THE END
NOW CLAP

Q/A and Contacts



sshcherbakov@plesk.com

twitter.com/stasian

t.me/xSTASiANx